REMARKS

Claims 1-20 remain pending in this application. No claims are amended.

Claim Rejections – 35 U.S.C. § 101

Claims 1-20 stand rejected under 35 U.S.C § 101. Respectfully we disagree.

Regarding claims 1-17, the claimed methods have statutory subject matter in that they, for example, transform binary data (chassis codes) into a human readable form (text strings). These text strings may be output to a log file if so requested by a user. See, for example, paragraph [0023] of the specification. Paragraph [0034] of the specification teaches "detected problems may be detailed for review by relatively un-trained persons."

This is not then just an "abstract" idea as alleged by the Examiner. The claims have use as a practical application wherein persons may review, for example, text (a tangible result) regarding chassis codes (processed, for example, through a getcc algorithm of a computing system); prior to the inventions of claims 1-17 these persons had to review cumbersome and difficult to read data.

Reconsideration of claims 1-17 is respectfully requested.

Claims 18-20 relate to a system for processing events from electronic architecture. As shown in FIG. 2, system 100 includes getcc processing section 102, analyzers 120 which, as described, are software tools. Therefore, system 100 is clearly a computer system with processing capabilities. As above, system 100 provides tangible, useful output by translating events (chassis codes) into human readable form (text strings).

Reconsideration of claims 18-20 is respectfully requested.

Claim Rejections – 35 U.S.C. § 102

Claims 1-4 and 6-20 stand rejected under 35 U.S.C. § 102(e) as being anticipated by US Patent Publication Number US2003/0074607 A1 to Brundridge et al. (hereinafter "Brundridge"). Respectfully, we disagree.

To anticipate a claim, Brundridge must teach every element of the claim and "the identical invention must be shown in as complete detail as contained in the ... claim." *MPEP 2131* citing *Verdegaal Bros. V. Union Oil Co. of California*, 814 F.2d 628, 2 USPQ2d 1051 (Fed. Cir. 1987) and *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 9 USPQ2d 1913 (Fed. Cir. 1989). Brundridge does not teach every element of claims 1-4 and 6-20.

In Applicants' response of February 23, 2004, and again in Applicants' amended appeal brief of March 23, 2005, Applicants explained that events are internally generated by

entities of electronic architecture. In paragraph [0008] of the specification, for example, event extraction is described in the context of an extraction tool that "connects with electronic architecture to *extract* and analyze *internally generated events*" (emphasis added). These events are thus generated independently of the extraction tool used to extract the events from the electronic architecture. These events, once extracted from the electronic architecture, are separated, filtered and transformed into text strings. For example, chassis codes may take the form of two 64-bit numbers (one number detailing system information, one number defining context sensitive information). See paragraph [0027] of the specification.

On the other hand, Brundridge discloses "an end-user with a problem acknowledges the problem and runs a diagnostic program directed to the particular device ... the diagnostic program returns a*[sic]* 'error-level' condition." See Brundridge page 2, paragraph [0025]. Brundridge must run a diagnostic to generate an error code. For each error code generated, "a keyword is placed in the error log after an error string along with an event code (error code)." See Brundridge page 2, paragraph [0026]. Brundridge then discloses that the error log is parsed by searching for the keyword; once located "the script will retrieve the data to the immediate right ... until a white space character is received." See Brundridge page 3, paragraph [0029]. Brundridge continues: "If the event code is a duplicate of one stored in the array, the script will ignore the redundant error code" and "Once the error log has reached the EOF, the array is used to build a FAQ list that is displayed to the end user." *Id.* Clearly, Brundridge must run a diagnostic program to generate error codes that are stored in a log file with embedded keywords; it then parses the log file to locate the error codes, which are then used to select frequently asked questions (FAQ) for presentation to the user. The error codes of Brundridge are simple identifiers, as opposed to bit encoded chassis codes of the immediate application, and are not specifically decoded, but are related to particular FAQ files. See Brundridge page 4, paragraph [0036].

Claim 1 recites a method for processing events from electronic architecture, the architecture of the type having one or more entities generating the events, and requires the following steps:

(A) extracting the events from the architecture;

(B) separating the events according to the entities; and

(C) transforming the events to one or more text strings.

Brundridge discloses a method "for detecting and reporting failures of devices in a computer system by assigning event codes to particular failures." See Brundridge paragraph

[0011]. Specifically, Brundridge detects failures of devices in a computer system and generates event codes based upon the failure. Brundridge does not disclose, at least, extracting the events from the architecture as required by step (A) of claim 1. As described in paragraph [0024] of the specification, and shown in element 218 of FIG. 3A, chassis codes are read from electronic architecture. These chassis codes are pre-defined codes generated by various entities within the electronic architecture. See paragraph [0002] of the specification.

Step (B) requires that the events extracted in step (A) are separated according to entity. Since Brundridge runs a diagnostic program directed to a particular device, the events of Brundridge relate to only one device. Therefore, Brundridge also cannot anticipate step (B) of claim 1.

Further, Brundridge's transformation of events into FAQs do not involve processes and features, or purpose, of element (C). Brundridge cannot therefore anticipate step (C) of claim 1.

For at least these reasons, Brundridge cannot anticipate claim 1; reconsideration is requested.

Claims 2 - 4 and 6-17 depend from claim 1 and benefit from like argument. However, these claims have other features that patentably distinguish over Brundridge. For example, claim 2 recites the step of filtering the events. As described table 1 of the present specification, the options –cell and –proc may be used to select which chassis codes to process and hence provide input to the filtering steps 230 and 232 of FIG. 3. Brundridge does not disclose such selective filtering; Brundridge merely discards duplicate events. See Brundridge page 3, paragraph [0030]. Brundridge does not allow selective filtering as with the immediate application.

Claim 3 recites extracting chassis logs, separating the chassis logs, and transforming the chassis logs to text strings. The Examiner asserts that the error logs of Brundridge are similar to chassis logs of the immediate application. We again disagree. The error codes of Brundridge are simple codes, whereas chassis codes are formed of two 64-bit numbers (one number detailing system information, one number defining context sensitive information). The chassis codes, as used in the immediate application, are generated by entities of the electronic architecture.

Claim 4 recites coupling a getcc extraction tool to the architecture. As previously argued, chassis codes are extracted from an electronic architecture. This is accomplished by the getcc extraction tool. As taught by paragraphs [0019-0028], getcc extracts chassis codes and processes them based upon configuration settings. Brundridge does not disclose such a

tool, and therefore, cannot anticipate claim 4. The parsing program written in JAVA of Brundridge does not extract events from an electronic architecture and, for at least this reason, cannot anticipate the getcc extraction tool of claim 4.

Claim 6 recites the architecture being a server, wherein the step of extracting events from the architecture comprises extracting events from the server. In view of claim 1, Brundridge cannot anticipate claim 6. For example, Brundridge does not extract events from servers.

Claim 7 recites converting a binary representation of the events to text strings. Brundridge does not extract events from electronic architecture, and does not disclose converting a binary representation of the events to text strings. The events of Brundridge are stored as alpha-numeric characters. See for example Brundridge page 2, paragraph [0026] and the exemplary error log therein.

Claim 8 recites analyzing the text strings and producing a human interpretable statement summarizing at least one of the events associated with the text strings. Brundridge does not disclose summarizing at least one of the events associated with the text strings. Instead, Brundridge discloses storing event codes retrieved from the log in an array which is used to build a list of frequently asked questions for display to a user. See Brundridge page 3, paragraph [0029]. This list of frequently asked questions is pertinent to the event code and is used to further isolate the problem and suggest a possible solution. See Brundridge page 1, paragraph [0005].

Claim 9 recites wherein the entities comprises one or more of firmware, software, processors, architecture monitors, power monitors, cabinet monitors, and I/O drivers. As argued above, Brundridge does not disclose extracting events from one or more entities.

Claim 10 recites controlling one or more steps of extracting, separating and transforming via one or more command line options. As argued above, Brundridge does not disclose extracting events from an electronic architecture. Contrary to the Examiner's assertion, Brundridge does not disclose command line options in paragraph [0028] of page 2. In table 1 of the immediate application, exemplary command line options are taught. For example, table 1 lists a command line option "–c" followed by a cell number to specify that chassis codes from the specified cell be processed; thereby controlling a filter that limits output of the getcc processing section 102. Brundridge discloses no such command line options.

Claim 11 recites controlling one or more steps of extracting, separating and transforming according to one or more configuration files. Contrary to the Examiner's

assertion, Brundridge makes no disclosure of a configuration file for controlling extracting, separating and transforming. Therefore, Brundridge cannot anticipate claim 11.

Claim 12 recites inputting the command line options via a graphical user interface. As argued above, Brundridge does not disclose command line options and cannot anticipate claim 12.

Claim 13 recites updating the command line options automatically from the architecture. As taught by paragraph [0021] of the specification, "getcc section 102 checks architecture 104 ... for any updates to itself." Brundridge discloses no such update.

Claim 14 recites specifying one or more cells of the architecture, and extracting the events only from the one or more cells. As argued above, Brundridge does not disclose extracting events from an architecture, and makes no mention of extracting events from specific cells of the architecture such as in claim 14.

Claim 15 recites specifying one or more processors of the architecture, and extracting the events only from the one or more processors. As argued above, Brundridge does not disclose extracting events from an architecture, and makes no mention of extracting events from specific processors of the architecture (see, e.g., table 1 of the present specification). Brundridge does not specify one or more processors of the architecture.

Claim 16 recites saving a log file representative of the events. As argued above, Brundridge does not anticipate claim 1 and therefore for at least this reason cannot anticipate claim 16.

Claim 17 recites transmitting the text strings to one or more analyzers associated with one or more entities and analyzing the text strings at the one or more analyzers. As shown in FIG. 2 and described in paragraph [0027] of the specification, one or more analyzers 120 analyze text strings from getcc section 102 in association with a specific entity (e.g., processor 20, I/O drivers 24, service processor 30, all of FIG. 1) of the architecture. Brundridge does not disclose distributing text strings to one or more analyzers associated with one or more entities for analysis. Therefore, Brundridge cannot anticipate claim 17.

Claim 18 recites a system for processing events from electronic architecture, the architecture of the type having one or more entities generating the events, including:

a) an extraction tool for extracting the events from the architecture, separating the events according to the entities, and transforming the events to one or more text strings; and

b) an interface for coupling the extraction tool to one or more of the architecture and a log file storing the events from the architecture.

As argued above, Brundridge does not disclose an extraction tool for extracting events from an architecture, separating events according to entities, or transforming the events into one or more text strings as required by element a) of claim 18. Brundridge discloses a diagnostic tool directed at a particular device that is causing problems. See Brundridge page 2, paragraph [0025]. The method of Brundridge requires that a user run a diagnostic tool to diagnose a problem device. Error strings returned by the diagnostic tool are then parsed to identify event codes that are then used to select FAQs to help the user identify the problem and suggest possible solution. Brundridge does not disclose an interface for coupling the extraction tool to one or more or the architecture and a log file for storing the events from the architecture as required by step b). Therefore, Brundridge cannot anticipate claim 18.

Claim 19 recites wherein the entities comprise one or more of firmware, software, processors, architecture monitors, power monitors, cabinet monitors, and I/O drivers, and wherein the events comprise chassis logs from one or more of the firmware, software, processors, architecture monitors, power monitors, cabinet monitors, and I/O drivers. As argued above, Brundridge does not disclose extracting chassis logs from multiple entities, and therefore cannot anticipate claim 19.

Claim 20 recites one or more analyzers coupled to the extraction tool, the analyzers processing the text strings into one or more human interpretable statements summarizing at least one of the events associated with the text strings. As argued above, Brundridge does not disclose one or more analyzers, coupled to the extraction tool, for processing text strings into one or more human interpretable statements summarizing at least one of the events associated with the text strings. Therefore, Brundridge cannot anticipate claim 20.

Claim Rejections – 35 U.S.C. § 103

When applying 35 U.S.C. §103, the following tenets of patent law must be adhered to:

a) The claimed invention must be considered as a whole;

b) The references must be considered as a whole and must suggest the desirability and thus the obviousness of making the combination;

c) The references must be viewed without the benefit of impermissible hindsight vision afforded by the claimed invention; and

d) Reasonable expectation of success is the standard with which obviousness is determined. MPEP §2141.01, *Hodosh v. Block Drug Co., Inc.*, 786 F.2d 1136, 1134 n.5, 229 U.S.P.Q. 182, 187 n.5 (Fed. Cir. 1986).

In addition, it is respectfully noted that to substantiate a *prima facie* case of obviousness the initial burden rests with the Examiner who must fulfill three requirements. **First**, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the references or to combine reference teachings. **Second**, there must be a reasonable expectation of success. **Finally**, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on Applicant's disclosure. (emphasis and formatting added) MPEP § 2143, *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

Claim 5 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over Brundridge in view of U.S. Patent Number 6,269,398 granted to Leong et al. (hereinafter "Leong"). Respectfully Applicants disagree.

As argued above Brundridge does not anticipate claim 1. Leong discloses a method and system for monitoring remote routers in networks. Leong does not disclose extracting events from entities of an electronic architecture and does not make up for the shortfall of Brundridge in anticipating claim 1. Therefore, even when combined, Brundridge and Leong do not render claim 5 obvious. Reconsideration of claim 5 is respectfully requested.

Applicants have addressed all issues raised in the Office Action dated June 15, 2005, and respectfully solicit a Notice of Allowance for claims 1-20. Applicants believe no fees are currently due, however, if any fee is deemed necessary in connection with this Amendment and Response, please charge Deposit Account No. 08-2025.

Respectfully submitted,

Date: September 15, 2005

By 

Curtis Vock, Reg. No.: 38,356
LATHROP & GAGE, L.C.
4845 Pearl East Circle, Suite 300
Boulder, Colorado 80301
Tele: (720) 931-3011
Fax: (720) 931-3001